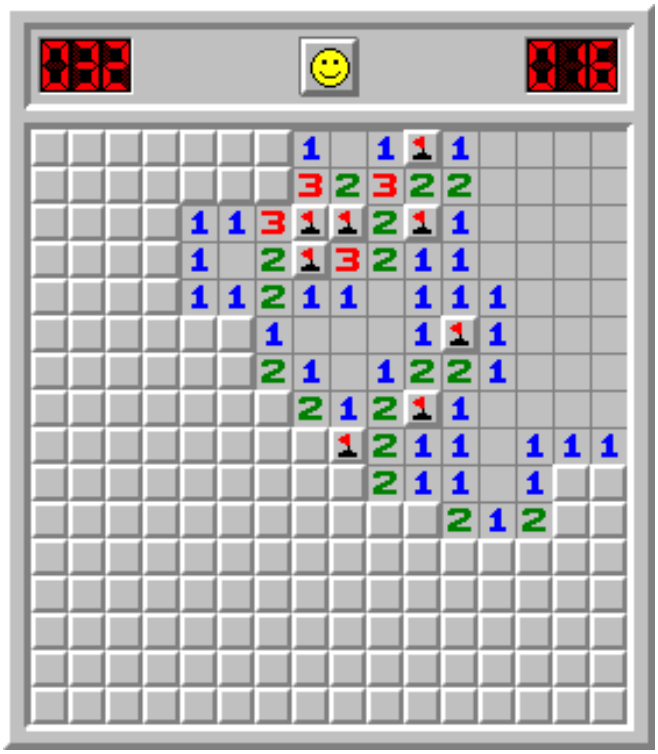
Minesweeper

Finding the optimal first move

By Arsalan Sepahpour



What is Minesweeper:

Minesweeper is a classic single-player puzzle game that was first included with the Microsoft Windows operating system in 1990. The game is played on a square grid, which represents a minefield. The player's objective is to clear the minefield without detonating any mines.

To play the game, the player must click on squares to reveal what is underneath. If a square contains a mine, the game is over, and the player loses. If a square does not contain a mine, a number is revealed that represents the number of mines in the surrounding eight squares. The player can then use this information to deduce which squares contain mines and which do not.

Using HTML, CSS, and JavaScript, I developed my own iteration of Minesweeper for this project. The grid in my version measures 5x5 and for each new game, I randomly select between 1-4 bombs to be placed at random locations within the grid.

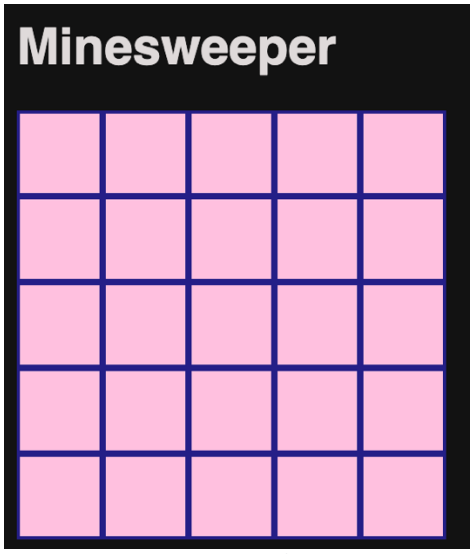


Figure 1: My version of Minesweeper

Initial plays:

At the beginning of the game, the objective is to gather as much information as possible about the bomb locations. Since no information is available about the bomb placements, a uniform distribution is assumed across all tiles. This means that the likelihood of a bomb being located under any given tile is 0.1.

To identify the best move, we can utilize entropy to pinpoint the tile that decreases unpredictability to the greatest extent. This involves examining the grid in 3x3 segments and calculating the entropy for the middle tile of each segment, assuming that the objective is to reveal the middle tile.

Figure 2: probability of the tiles, with a 3x3 window

Given the lack of initial information about the grid, every 3x3 grid possesses identical entropy, as each tile carries equivalent probabilities. To select the optimal move among these possibilities, we need to contemplate our subsequent actions prior to executing the current one. It is crucial to determine which tile will yield the most information based on the concurrent selection.

To do this, we can look at the decision move as a tree with three outcomes.

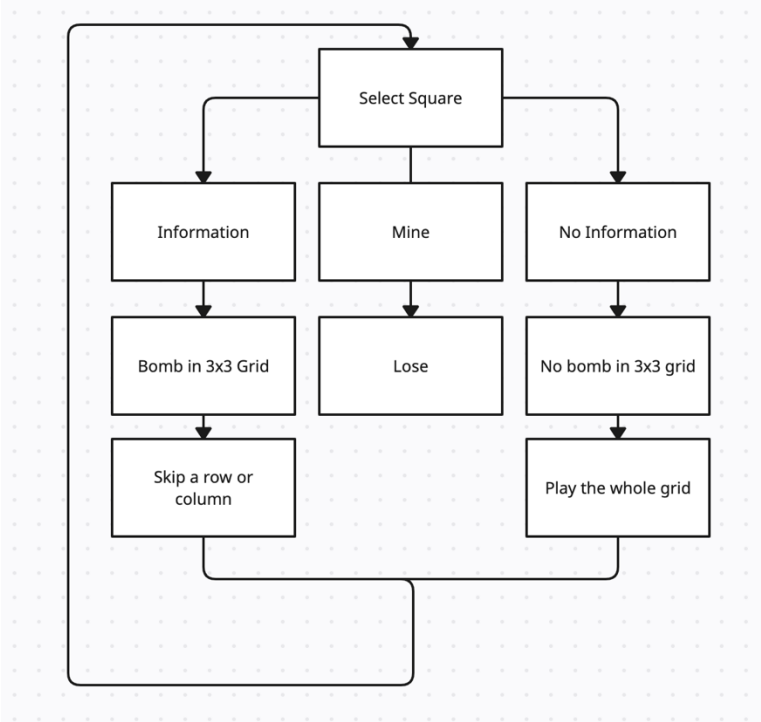


Figure 3: Decision tree for initial move

If the initial move reveals a mine, the user unfortunately loses. However, when faced with the two remaining outcomes of either obtaining or not obtaining information, we can make an

optimal initial move by computing the entropy of the entire grid after two moves. To find the best initial move.

To experiment to find the best initial move, I will set the number of bombs to 1 so its consistent with every iteration. Since the grid is symmetric, we can just search to find the best starting position with half of the grid.

The initial entropy of the game being:

$$H(G) = \sum_{i=1}^{25} p_i \log\left(\frac{1}{p_i}\right) = 25(0.04 \log(25)) = 4.64$$

The following positions are the ones I will find the entropy with.

Figure 4: First Test

In our first test, suppose we play the center tile and we get no information. The result will be the following.

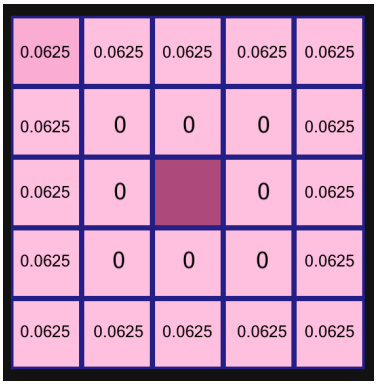


Figure 5: Center tile play with no information

With the updated probabilities we can calculate the entropy of the board again.

$$H(G) = \sum_{i=1}^{16} p_i \log\left(\frac{1}{p_i}\right) = 16(0.0625 \log(16)) = 4$$

Although this might be consistent with all the other combinations, its important to note that for at least one of the 0 probability squares there will be information, because there is only one block surrounding the whole grid. As shown below:

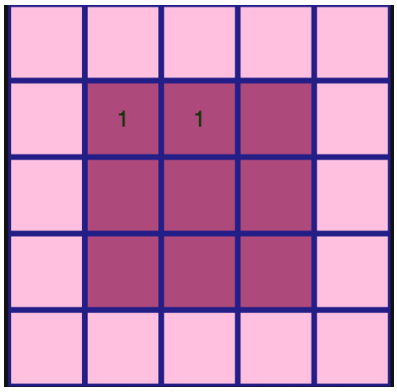


Figure 6: Play of zero probability boxes.

After the play of all the zero probability squares, we can narrow down the whole game to just two squares and one square in the following matter: (red box represents the probability of bomb being in that square)

- One square has information:

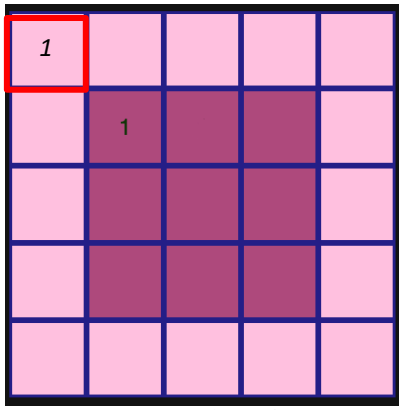


Figure 7: One square has information

With one square having information we can say with certainty that the mine is in the left corner, hence having a entropy of zero.

- Two squares have information:

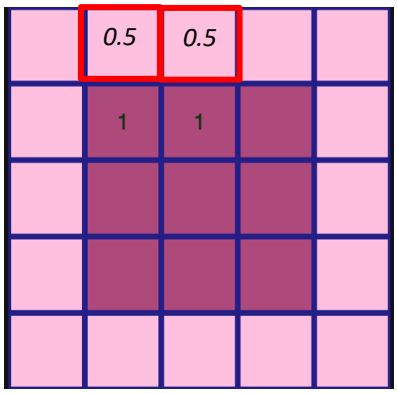


Figure 8: Two square information

If we get two squares with information, we reduce the entropy down to

$$H(G) = \sum_{i=1}^{2} p_i \log\left(\frac{1}{p_i}\right) = 2(0.5\log(2)) = 1$$

Where with just 1 move we can find where the mine is hiding.

- Three squares have information:

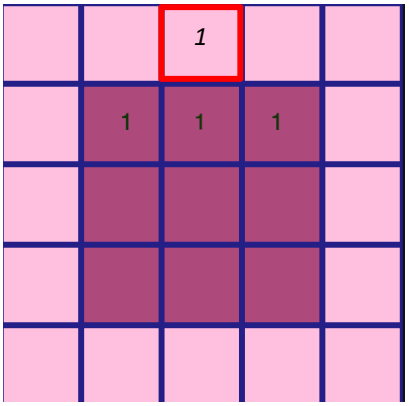


Figure 9: Three squares with information

Now for the second iteration we can go through this process using the middle upper square:

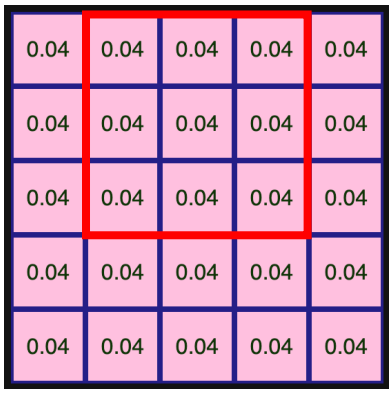


Figure 10: Second Test

The first play being the center square we can see entropy stays the same as the center square of the first play (figure 5):

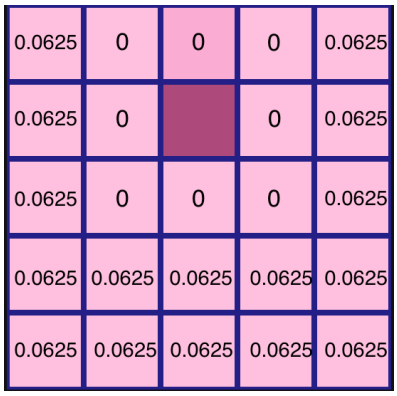


Figure 11: Center play of upper square

Getting to result to an entropy of:

$$H(G) = \sum_{i=1}^{16} p_i \log\left(\frac{1}{p_i}\right) = 16(0.0625\log(16)) = 4$$

The entropies all branch off after this play, contrary to the figure 5 play, there will not lie information for certain under the 0 probability squares.

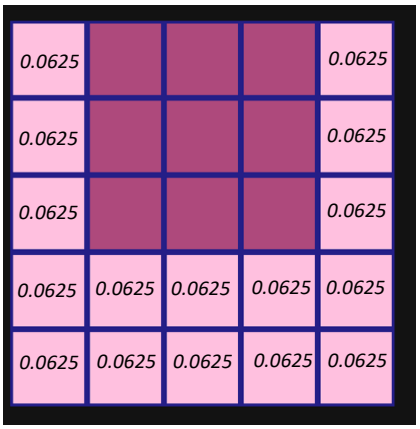


Figure 12: Second Test plays

As we can see, with more gameplays the entropy of the game does not change, hence the upper square move does not reduce the randomness of the game at all unless mine lies in the adjacent squares.

$$H(G) = \sum_{i=1}^{16} p_i \log\left(\frac{1}{p_i}\right) = 16(0.0625 \log(16)) = 4$$

Hence, we can conclude that the first play (figure 5) is a more optimized play compared to this.

Now we are down to the last play of which is the left corner.

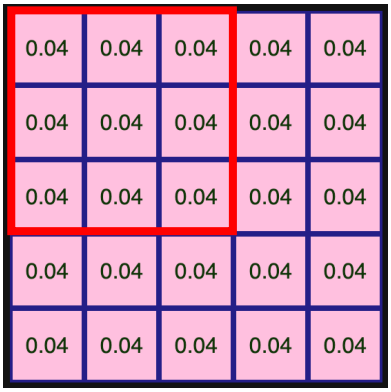


Figure 13: Third Test

The third test produces very similar results with the second test.

The first play being the center of our kernel with no information:

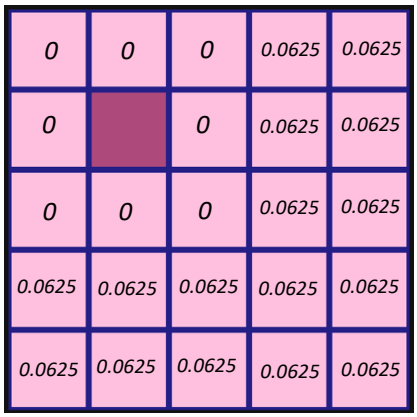


Figure 14: First play of left corner

The first play results in the same entropy as the first two plays.

$$H(G) = \sum_{i=1}^{16} p_i \log\left(\frac{1}{p_i}\right) = 16(0.0625 \log(16)) = 4$$

The following plays also product the same result as the second play:

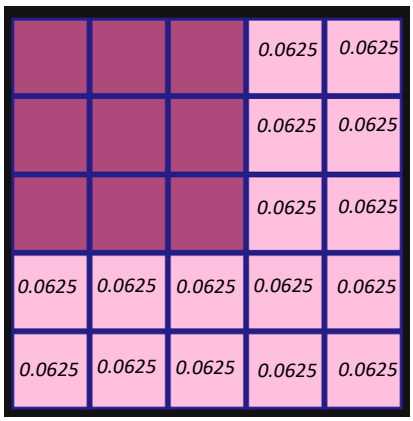


Figure 15: Playing all the left corners

Resulting again with the same entropy:

$$H(G) = \sum_{i=1}^{16} p_i \log\left(\frac{1}{p_i}\right) = 16(0.0625 \log(16)) = 4$$

Showing that by using the same gameplay as play 1, the entropy does not change at all.

Conclusion:

After examining the entropies of all possible starting moves, it was discovered that the initial move of choosing the center 3x3 resulted in the most significant reduction in randomness. This move allowed the bomb to be found in the fewest number of attempts. While a 5x5 grid with one bomb was used for this project, the same strategy could be implemented for larger grids with more mines by replicating the moves used previously, and applying the decision tree (figure 3) .

Github for developed game:
https://github.com/sepah99/Minesweeper